FairNet: A Genetic Framework to Reduce Marginalization in Social Networks

Federico Mazzoni[1,2], Andrea Failla[1,2], and Giulio Rossetti[2]

[1] University of Pisa, Department of Computer Science, Pisa 56127, Italy
`name.surname@phd.unipi.it`
[2] National Research Council, ISTI-CNR, Pisa 56127, Italy
`name.surname@isti.cnr.it`

Abstract. Discrimination in social networks often assumes the form of *marginalization* against nodes with specific features, e.g., segregation of/against minorities. In this work, we propose a metric that proxies social discrimination based on salient node features in a social network. Under the assumption that in a fair social system, all individuals should be enclosed in similar social circles representing the network in its entirety, our metric assigns a *marginalization score* to each node in the network, identifying if they are marginalized by similar nodes (e.g., a man marginalized by other men), by different nodes (e.g., a man marginalized by women), or not marginalized at all (i.e., the node has a fair neighborhood). Moreover, we introduce FairNet, a two-fold framework that aims to reduce network marginalization in partially- and fully-attributed networks by employing genetic algorithms. We evaluate our framework on networks emerging from online social interactions and find that the two components of FairNet are able to consistently reduce marginalization.

Keywords: Fairness · Genetic Algorithm · Marginalization · Bias Mitigation

1 Introduction

The study of social discrimination and inequalities has long benefitted from network analysis tools. Methods from network science can characterize a wide variety of phenomena that induce and/or foster social exclusion and community segregation, among gender inequalities in science [13], racial discrimination in online social networks [24], and weight stigma in adolescents [2]. Moreover, they can quantify the structural effects of human and algorithmic biases [23], and minority groups underrepresentation [16]. Inspired by recent literature in Artificial Intelligence and Machine Learning focusing on algorithmic discrimination, the study of *algorithmic fairness* is a rising trend in Network Science [29]. Fair Social Network Analysis is concerned with individuating fairness constraints on network-based tasks, and developing algorithms and tools that enforce/abide by such constraints. Another line of work focuses on the marginalization that might be endured by minorities, e.g. their limited access to information [35]. This sort of marginalization can be conceptualized as segregation within a node's social circle, or neighborhood. In other words, people belonging to a certain group might be marginalized by people belonging to another group, e.g., people with different skill levels [6] or ethnic background [10]. With this work, we propose a marginalization metric related both to the notion of *proportional fairness* proposed for clusters [8], and to the aforementioned idea of segregation. We assume that, in a fair system represented as a network with node attributes, a node has a "fair" neighborhood if the distribution of nodes' labels in that neighborhood is proportionally similar to the labels' distribution in the entire network. In other words, the diversity in a node's neighborhood should be representative of the diversity in the entire network. To mitigate marginalization in a social network and increase its overall fairness we introduce FairNet, a framework consisting of two algorithms: FairEdges covers the case of partially attributed networks, filling in missing metadata with values that lead to fairer neighborhoods, whereas FairLabels reduces discrimination via an edge addition strategy.

The rest of this work is organized as follows. Section 2 sums up relevant literature on the main topics surrounding this work. in Section 3, we provide an overview of the novel discrimination metric we introduce. This is discussed in detail in Appendix 5. Subsequently, we describe our framework to reduce the network's marginalization. In Section 4, the algorithms are tested on real-world data. Finally, Section 5 reports the conclusions and potential future works.

2 Related Work

In the following, we provide an overview of the main topics related to this work. Specifically, we first introduce the main concepts related to algorithmic fairness. Then, we move on to recent advances in fair social network analysis. Lastly, since the methods we propose in this work are based on genetic algorithms, we also provide an introduction to the matter.

Fairness. Two broad definitions of "fairness" have been proposed in the Data Science literature, namely *group fairness* and *individual fairness.* The group-based notion of fairness suggests that each group should have a similar proportion of individuals receiving the same treatment. For instance, if men are hired at a rate of 80%, then women should also be hired at approximately the same rate. Groups are defined following the value of *sensitive attributes* such as *Gender* or *Race.* This notion of fairness is quantified by metrics such as Statistical Parity and Disparate Impact, which check the difference or the ratio between the positive rates of the privileged and discriminated classes [1] (e.g., the percentage of men and of women receiving a positive outcome). Group fairness is often employed in classification tasks, including Graph Mining applications [11], and enhanced by pre-processing (modifying the initial training set of data, aiming to create an "ideal world dataset" [30]), in-processing (employing classifiers with a built-in concept of fairness), or post-processing techniques (modifying some of the decisions taken by the algorithm). On the other hand, *individual fairness* assesses that *similar individuals should be treated similarly*, with ad-hoc definitions of similarity taking into account various individuals' attributes other than sensitive attributes [26], i.e., individual with a very similar profile should receive the same treatment. The *proportional fairness* proposed in [7] for clusters is an example of individual fairness [8]. Inspired by fair resource allocation studies, this proportionality-based notion of fairness aims to find centroids to create fair, proportional clusters. The centroid could represent a public facility, e.g., a park; in this example, an unfair distribution of centroids could prevent the highest possible amount of people from accessing the facility [7]. This method for fair clustering has been extended to graphs [22].

Fairness in Social Network Analysis. When applied to the study of complex network analysis, the term "fairness" refers to the research area concerned with developing tools and algorithms to enforce fairness constraints in network analysis tasks. Tsioutsiouliklis and colleagues proposed two feature-aware variants of the PageRank ranking algorithm that take into account group fairness constraints [33], and thus bump up the visibility of minority nodes. Similar reasoning is followed by [28], where the authors propose a network counterpart of the Statistical Parity metric, checking if each group has a similar acceptance rate while sending a friend request. Another metric proposed in the same work is *Equality of Representation*, checking if each group has the same visibility (e.g., the same chances of featuring in friend suggestions). Stoica and Chaintreau proposed solutions to the influence maximization problem while accounting for fairness both in the selection of seed nodes, and in the outreach of the diffusion process [32]. A conspicuous body of works also relates to bias and discrimination in social networks while not mentioning fairness explicitly. For instance, [21] highlights bias towards lowly connected nodes in popular community detection methods. Moreover, strong homophilic or heterophilic behaviors in a network were proved to directly influence the degree ranking of nodes belonging to minorities [14], which can, in turn, limit their ability to access information [35]. This can be an issue for researchers, too, as (especially strong) differences in relative class sizes (e.g., a male majority vs. a female minority) can drive biases in analytical settings, e.g., in network sampling [34] and mixing patterns estimation [15].

Genetic Algorithms. Genetic Algorithms (henceforth *GA*) are a kind of metaheuristic algorithm commonly used for optimization problems (e.g., the *knapsack problem* [31]). A GA attempts to find the optimal solution by first generating a set of potential solutions and then mating and mutating them, resulting in a second generation of potential solutions. The best individuals of the new generation are selected with respect to a fitness function. These steps are repeated for a fixed amount of generations [17]. GAs have been employed to increase the fairness of tabular data; *AuFair* [36] makes use of GAs to create a fair set of rules to replace biased human-made decisions, while *GenFair* [20] to generate the synthetic data needed to balance a dataset following multiple fairness criteria.

In the context of networks, GAs have been used to solve various tasks, including community detection [27], information spread [5], and link prediction [4].

3 Methods

According to the notion of individual fairness, *similar individuals should receive similar treatment.* If we assume that similar individuals are those enclosed in the same system (i.e., nodes in the same network), then all of them should be treated similarly for the system to be considered fair. Taking into account individuals' characteristics determined by a sensitive attribute (e.g., gender), all nodes in the same network should be surrounded by a group of peers manifesting a similar attribute distribution. Additionally, such distribution should be representative of the label distribution in the entirety of the system. A proportionally different distribution would indeed imply some degree of discrimination against the node – either by nodes with different labels or by those with similar labels.
In the following, we first present a new metric, called *Individual Marginalization Score* (IMS), which can be exploited to proxy a node's marginalization. Subsequently, from the IMS of every individual in a system, we compute the overall *System Marginalization Score* (SMS), i.e., the marginalization score of the entire network. Concluding, we introduce the two algorithms constituting the FairNet framework, that improve the network's fairness as evaluated by these metrics.

3.1 Quantifying Marginalization

Let $G = (V, E, L)$ be a connected node-attributed graph where V is the set of nodes, E is the set of edges, L is the set of categorical node labels l such that l_v identifies the attribute value of node v, with $v \in V$. We denote with $\mathcal{N}_v$ the set of v's first-order neighbors, i.e., the nodes it has a direct connection to. $\mathcal{N}_v$ can be seen as the union of two disjoint sets: $\mathcal{N}_v^l = \{u : u \in \mathcal{N}_v \wedge l_u = l_v\}$ (i.e., the subset of nodes in $\mathcal{N}_v$ with the same label as v), and $\mathcal{N}_v^{\neg l} = \{u : u \in \mathcal{N}_v \wedge l_u \neq l_v\}$ (i.e., the subset of nodes in $\mathcal{N}_v$ with a different label than v's). Of course, at most one of $\mathcal{N}_v^l$ and $\mathcal{N}_v^{\neg l}$ can be empty. To quantify marginalization at a local scale, we introduce the Individual Marginalization Score (IMS), which can be computed for v as:

$$\mathbf{IMS}(v) = \frac{2|\mathcal{N}_v|\omega_{l_v}}{|\mathcal{N}_v^l|\omega_{l_v} + |\mathcal{N}_v^{\neg l}|(1 - \omega_{l_v})} - 1, \tag{1}$$

In this formula, ω_{l_v} is a weight assigned to l_v that depends on its distribution in the whole network. A more detailed description of how the weights are computed, as well as of the formula's rationale can be found in Appendix 5.
The formula returns:

i. $\mathbf{IMS}(v) > 0$ if $\mathcal{N}_v$ contains more nodes with the same label as v, which might imply some form of marginalization of v by nodes with a different label;

ii. $\mathbf{IMS}(v) < 0$ if $\mathcal{N}_v$ contains more nodes with a different label than v's, which might indicate some form of marginalization of v by nodes with the same label;
iii. $\mathbf{IMS}(v) = 0$ iif $\mathcal{N}_v$ has a fair distribution of nodes' labels. As the weights are computed w.r.t. the whole network, a score of 0 indicates that the label distribution in v's neighborhood is proportionally similar to that of the entire network.

In cases where node attributes can have more than two values, a *one vs. all* scenario is supposed, where the number of labels similar to v's is compared to the cardinality of the heterogeneous group of different labels. For instance, assuming v's *Race* value is `Black`, a neighborhood with 2 Black nodes, 3 White nodes, and 2 Asian nodes will have the same marginalization score as a neighborhood with 2 Black nodes 1 White node, and 4 Asian nodes.

The IMS expresses the marginalization at the node level. To quantify the marginalization of a system of individuals (i.e., a network), we propose the *System Marginalization Score* (SMS). The SMS ranges in [0, 1] and is computed as the mean of the absolute values of each node's marginalization score. Formally:

$$\mathbf{SMS}(G) = \frac{1}{|V|} \sum_{v \in V} |\mathbf{IMS}(v)|. \tag{2}$$

Alternatively, a network's overall discrimination can be quantified by counting the amount of discriminated nodes, setting a marginalization threshold τ; only nodes with the absolute value of their IMS above this threshold are considered discriminated. For instance, supposing $\tau = 0.7$, nodes with a score in the ranges $[0.7, 1]$ and $[-1, -0.7]$ are considered as discriminated. Note that this rationale assumes that each node is associated with an attribute value. In order to compute the IMS, nodes with missing labels should either be removed or treated.

3.2 Genetic Algorithm

In the next sections, we describe the two algorithms we devised to reduce a network's marginalization. Both of them employ a genetic algorithm, a search heuristic based on the concept of natural selection, or “survival of the fittest", as proposed by Darwin. This algorithm is frequently employed to tackle optimization problems. Each potential solution is referred to as a *chromosome* or *individual*, and it contains various “genes” [17]. The process begins with the generation of a random population of chromosomes, including p individuals. Each chromosome is then evaluated by a *fitness function*, which is closely tied to the problem to be solved.

Following this, the *selection operator* of the GA picks the optimal individual in the population with the goal of either minimizing or maximizing the fitness value. Essentially, only the “fittest” individuals are allowed to survive. A frequently used selection operator is *Tournament selection*, which randomly selects a number of chromosomes to participate in a tournament. The chromosomes with the highest fitness are declared the winners [17]. The selected chromosomes then have a chance c to be combined (usually in pairs) by the *crossover operator*. Selected genes are "shuffled" between each chromosome, resulting in new "children" individuals that inherit their genes from the "parent" individuals. The offspring population is referred to as the *second generation* of individuals. Offspring may be chosen by a *mutation operator*, following a probability m, which results in at least one of their genes being mutated.

This process is then carried out for k generations, with chromosomes from newer generations evaluated, selected, and potentially mated and mutated. As the algorithm retains only the fittest individuals, the overall fitness tends to improve with each generation. In the end, the algorithm returns the individual(s) with the highest fitness value as the optimal solution(s) to the given problem.

In Algorithm 1, we report the pseudo-code of the archetypical GA – including the default values employed by our algorithms. Please note that, in practice, evaluation functions and more advanced operators might require additional parameters (as is the case for both FairLabels and FairEdges).

3.3 FairLabels

In this section, we introduce FairLabels, an algorithm that replaces nodes' missing metadata with values that reduce the overall marginalization. There are essentially two strategies when it comes to working with partially attributed networks, namely (i) removing nodes without metadata or (ii) filling in missing values. While approach (i) is the most straightforward — and perhaps the most common — removing entities from the network can lead to non-negligible information loss. This is especially true for networked data since node deletion also implies the elimination of all of its links, potentially leading to a significant alteration of the topology. Moreover, there is an inherent selection bias in removing data points with missing information, especially in mining tasks. As such, this pre-processing step is of critical importance.

Our proposal aims to tackle the issue of missing node metadata by inserting values in a way that leads to a fairer network. Note that our aim here is not to solve a node label prediction problem, but rather to find a combination of potential labels so as to minimize the number of discriminated nodes in the network (which is an optimization problem a GA can handle). The algorithm's rationale[3] is outlined in Algorithm 2. First, FairLabels identifies M, the set of all the nodes with a missing label (line 1). In line 2, M is given as input to a genetic algorithm alongside the network G, τ, and the parameter a (see below). Each individual created by the genetic algorithm has the same cardinality as M (i.e., each gene represents a node with a missing label). The value of each gene is a possible value from L; e.g., `Black`, `White`, `Asian` for the *Race* attribute. In other words, each individual tested by the genetic algorithm is a vector of labels $l \in L$ for

---

[3] It should be noted that both FairLabels and FairEdges also take as input the GA parameters described in Section 3.2. For the sake of simplicity, they were not included in the respective pseudo-codes, nor in the overview of the two algorithms.

Algorithm 1: Genetic Algorithm

```
Input  : $p$ - population size, default: 150
         $c$ - crossover probability, default: 0.5
         $m$ - mutation probability, default: 0.25
         $t$ - tournament size, default: 3
         $k$ - number of generations, default: 100
Output:  $s$ - best solution found
$best \leftarrow [\,]$;
$gen \leftarrow 0$;
$pop \leftarrow init(P)$;                                  // initial population
for $individual$ in $pop$ do                               // initial evaluation
    $eva \leftarrow evaluate(individual)$;
    $best$.append($(eva, individual)$)
while $gen < k$ do
    $best \leftarrow select(best,\ t))$;                   // initial selection
    $offspring \leftarrow [\,]$
    for $parents$ in $best$ do                             // crossover
        if $random(0,\ 1) < c$ then
            $children \leftarrow crover(parents)$;
            $offspring$.append($(children)$)
    for $child$ in $offspring$ do                          // mutation
        if $random(0,\ 1) < m$ then
            $child \leftarrow mutate(child)$
        $eva \leftarrow evaluate(individual)$;
        $best$.append($(eva, individual)$)
    $best \leftarrow select(best,\ t))$
    $gen \leftarrow gen + 1$
$s' \leftarrow picking\_best(best)$;                       // picking the best result
return $s$
```

the nodes. Fig. 1 provides a visual representation. Each solution is evaluated by inserting the labels in G and then computing the number of discriminated nodes in the network[4] w.r.t. τ. The best chromosomes are then mated, and their children are possibly mutated. As crossover operator, we employed the traditional Two Point Crossover. The operator selects two random "parent" chromosomes, and in each of them, two genes that are in the same position. The genes between the selected ones are then shuffled between the parents, resulting in two new "children" chromosomes [18]. For the mutation, we devised a custom operator. If the mutation operator picks a child chromosome (as outlined in Section 3.2), the value of each gene has a probability a of mutating into another (default:

---

[4] Preliminary experiments showed that the genetic algorithm performed better with the number of discriminated nodes rather than with SMS. A minimal increment in SMS – resulting nonetheless in the same number of discriminated nodes – is seen as an improvement by the GA.

Algorithm 2: FairLabels

```
Input  : $G$ - graph with missing labels
         $\tau$ - marginalization threshold
         $a$ - probability of gene mutation
Output: $G'$ - graph without missing labels
1 $M \leftarrow missing\_check(G, \tau)$;          // extract the set of nodes with missing labels
2 $G' \leftarrow label\_genetic\_algorithm(G, M, \tau, a)$; // find the best combination of values
    for the missing labels
3 return $G'$
```

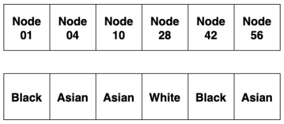

Fig. 1: Starting from a vector of nodes with missing labels, FairLabels can fill chromosomes with the label values (e.g., *Race* values).

0.05). The best solutions are then selected by conducting a Tournament Selection. These steps are repeated for a set amount of generations; afterward, the algorithm returns G', the network filled with the best combinations of labels found.

3.4 FairEdges

FairNet's other component is FairEdges, an algorithm to reduce the network's marginalization by adding new edges. We report the pseudo-code in Algorithm 3. FairEdges takes as input G, τ, and p (the latter representing the percentage of edges to be considered viable; see below), and it is structured in three phases. After computing (in line 1) the network marginalization as outlined above, in line 2, FairEdges selects a set of edges to be added according to the sociological principle of triadic closure [12], see below. In line 3, FairEdges then employs a genetic algorithm to find the best combination of edges to add so as to reduce the network's discrimination while operating the least amount of topological changes. These edges are added to G, resulting in a fairer network G', which is returned to the user. FairEdges has some similarities with the fair clustering approaches mentioned in Section 2. However, the clustering algorithms aim to find the best "core" (i.e., the best centroid) to create fair sets of points, whereas FairEdges, given a core, finds the best set of edges. Fig. 2 provides a visual representation.

Edge Selection. After computing marginalization scores for all nodes via Eq. 1, we consider an individual to be marginalized if its IMS is positive and higher, or negative and lower, than a fixed threshold τ. The edge selection phase, detailed in Algorithm 4 aims at constructing a set of edges E' that improve the condition

Algorithm 3: FairEdges

```
Input  : $G$ - graph,
         $\tau$ - marginalization threshold,
         $p$ - edge percentage
Output: $G$ - balanced graph
$V' \leftarrow marginalization\_check(G, \tau)$;                 // find marginalized nodes
$E' \leftarrow edge\_selection(G, V', p)$;                          // see Algorithm 4
$best \leftarrow edge\_genetic\_algorithm(G, E', \tau)$;   // find best combinations of edges
$G$.add($best$)
return $G$
```

Algorithm 4: Edge Selection

```
   Input  : $G$ - graph,
            $V'$ - marginalized nodes,
            $p$ - edge percentage
   Output: $E'$ - selected edges
1  $E' \leftarrow$ find_nonexisting_edges($G$, $V'$)
2  for $e \in E'$ do
3      e.weight = compute_triangles($e$, $G$);
4  $E' \leftarrow$ sort_by_weight($E'$)
5  $n \leftarrow$ length($E'$);
6  $k \leftarrow$ int($p \times n$)                                   // calculate the top $p\%$
7  $E' \leftarrow E'[1 : k]$                                          // select the top $k$ edges
8  return $E'$
```

of such marginalized individuals. As a first step, the algorithm collects all non-existing edges adjacent to at least one marginalized node (as defined above). For the network modifications to be realistic, we need a way to estimate the plausibility of non-existing ties. Indeed, in real-world networks, it is often the case that not all connections are equally important – and we would not want to add edges that are unlikely to appear. In principle, this can be done with any heuristic commonly employed in link prediction tasks [19]. In this work, we use triadic closure as a proxy for tie strength/plausibility. Thus, we assign each edge a weight based on the number of triangles it would close if it were to be added to the network (lines 2-3). In other words, non-existing edges that would close *more* triangles are considered more likely to appear than those with a lower weight. Then, the algorithm sorts possible edges in descending order of weight (line 4). Subsequently, only $p\%$ edges with the highest weights are kept, while others are discarded (lines 5-7). This is done both to reduce implausibility and runtime.

Minimizing Marginalization FairEdges conceptualizes minimizing the network's marginalization as an optimization problem; given the set of edges E', FairEdges finds the best combination of edges to be added to reduce the marginalization. FairEdges's GA is similar to that of FairLabels, albeit more

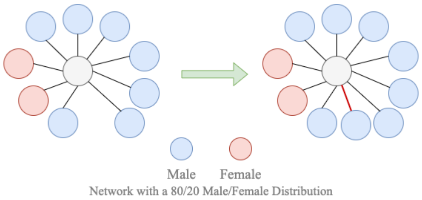

Fig. 2: Visual representation of FairEdges. Note that, since the IMS does not depend on the label of the core node, but rather on those of its neighbors, FairEdges's solution is oblivious to the node's label. Indeed, in case the core node were male, it would be marginalized by its peers, and FairEdges would add an additional male node to its neighborhood. Conversely, if the core node were female, it would be marginalized by nodes with a different label, thus still leading to the addition of a new male node.

A ↔ B	C ↔ B	E ↔ H	C ↔ G	A ↔ G	A ↔ H
1	0	1	1	0	0

Fig. 3: Starting from a vector of edges to be added, FairEdges creates binary chromosomes to find the optimal solution. Each 1 in a chromosome results in a modification to the original network structure (i.e., the corresponding edge is created). Conversely, each 0 implies no modification will concern the corresponding edge (i.e., the edge is not created).

complex due to the task. The GA creates P vectors of binary values S with $|S| = |E'|$ as the initial chromosomes. Each of their genes represents an edge in E'; if it has a value of 1, then the possible solution modifies the corresponding edge in G, whereas with 0, no alteration is carried out. Fig. 3 provides a visual representation.

Our fitness function alters G by adding edges represented by a 1. As in FairLabels, we employed the number of discriminated nodes as the fitness metric. If two possible solutions result in the same network's discrimination, the most conservative solution has priority (i.e., the solution with fewer 1s). Two Points Crossover and Tournament Selection are again employed. Given that the value of each gene can be either 1 or 0, we used the traditional Flip Mutation as our mutation operator. It mutates a gene 0 value into a 1, and vice-versa. The steps are then repeated for the set number of generations. The network modified by adding the best set of edges (as found by the GA) is then returned to the user.

4 Experiments

We tested FairLabels and FairEdges on two networks. The *Twitter Network*, introduced in [3], features 3,753 nodes, connected by 6,993 edges. The sensitive

Table 1: Comparison between the number of discriminated nodes either by removing or filling missing labels.
Each test was performed 50 times; values in brackets are the standard deviation.

Network	Attribute	Nodes	Edges	Missing	Thresh.	Removing	Filling
Twitter	*Inclination*	3,753	6,993	375	0.3	760 - 22%	862.86 (7.86) - 23%
					0.5	659 - 20%	700.48 (6.59) - 19%
					0.7	539 - 16%	506.68 (4.42) - 14%
Facebook	*Gender*	5,180	186,586	418	0.3	1,563 - 32%	1,434.32 (8.02) - 28%
					0.5	586 - 12%	461.64 (3.11) - 9%
					0.7	204 - 4%	154.34 (1.19) - 3%

Table 2: Comparison between the number of discriminated nodes by applying FairEdges or randomly adding edges.
Each test was performed 50 times; values in brackets refer to the standard deviation.

Network	Attribute	Nodes	Edges	Thresh.	Disc. Nodes	Benchmark	Final Result	Added Edges
Twitter	*Inclination*	3, 753	6, 993	0.3	1,017	645.56 (4.15)	589.84 (7.16)	2,361.52 (37.28)
				0.5	863	525.4 (3.12)	472.74 (5.75)	1,083.78 (20.41)
				0.7	701	484.62 (2.92)	437.06 (4.30)	389.20 (11.29)
Facebook	*Gender*	4,762	169,128	0.3	1,563	611.64 (3.04)	572.48 (6.84)	30,310.52 (120.5)
				0.5	586	190.52 (2.17)	164.24 (3.37)	2,811.58 (29.4)
				0.7	204	146.4 (1.13)	137.94 (0.64)	132.3 (16.36)

attribute is *Political Inclination* and can assume three values: *neutral* (2,598 nodes), *liberal* (782 nodes), and *conservative* (180 nodes). The *Facebook Network* has 5,180 nodes, including 418 with missing labels, connected by 186,586 edges. If nodes with missing values are removed, the network is left with 169,128 edges and 4,762 nodes. The binary sensitive attribute *Gender* can be *Male* (2,153 nodes) or *Female* (2,609 nodes). Our algorithms were tested with their default GA parameters and three τ values: 0.3, 0.5, and 0.7.

4.1 Testing FairLabels

First of all, we tested FairLabels. As said, the Facebook network has 418 missing labels. The Twitter network featured no missing labels; therefore, 10% of the nodes (375) were randomly selected for these experiments and had their label removed. Table 1 compares the initial number of discriminated nodes if the missing values are just removed, or if they are filled by FairLabels. We featured both raw and relative frequencies of discriminated nodes. Each test was run 50 times; the *Filling* columns report the average result and the (generally low) standard deviation. With the Twitter network, removing or filling missing labels generally leads to comparable results, although the filling option is slightly better. This is noticeable by looking at proportional values. Nonetheless, employing FairLa-

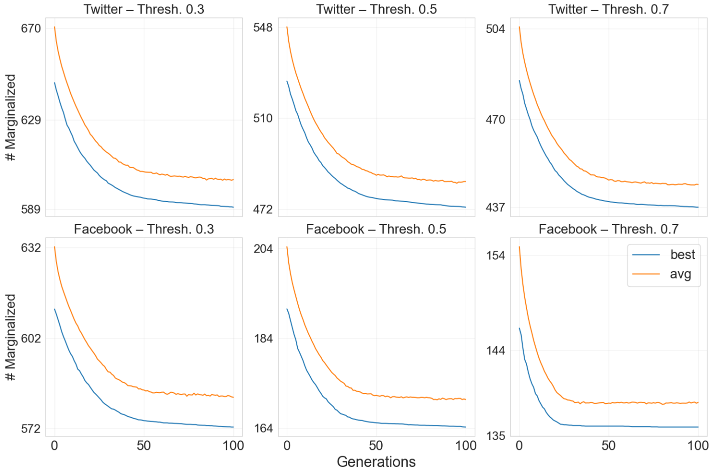

Fig. 4: Best (blue) and mean (orange) values for each generation in FairEdges's genetic algorithm. Each value is averaged over 50 runs.

bels is by design less intrusive than outright removing nodes, as no information from the original network is lost. As for the Facebook network, FairLabels always leads to a fairer initial state.

4.2 Testing FairEdges

Table 2 reports the results of our tests. Since we focused on FairEdges alone, for these experiments, the nodes with missing labels in the Facebook Network were removed and not replaced. As expected, increasing the marginalization threshold decreases the number of discriminated nodes. Again, each test was run 50 times. The *Benchmark* column reports the average of the best results obtained during the 0th generation of the genetic algorithm, i.e., the result obtained by adding a random selection of links. By contrast, the *Final Result* column reports the average of the best results at the end of the final, 100th generation. The *Added Edges* column reports the average number of added edges to achieve the best result.

As seen in the *Benchmark* column, randomly adding edges (selected from the edges pool described above) already greatly decreases the number of marginalized nodes. The genetic algorithm, however, always manages to further increase the network's fairness, to an extent never reached by random selection. This is more noticeable with higher amounts of discriminated nodes. We noticed a sharper decrease in discriminated notes during earlier generations, as the results obtained by the genetic algorithm were close to a convergence around the 50th generation (see Fig. 4).

5 Conclusion

In this work, we presented a new metric to measure marginalization in networks, inspired both by the concept of individual fairness and by the works on segregation in networks. Our metric assumes that in a fair network, the label distribution in a node's neighborhood should mirror the label distribution in the entire network. Furthermore, we have proposed the FairNet framework, including two algorithms following a genetic approach to reduce the number of discriminated nodes w.r.t. the aforementioned metric. Experimental results prove that both FairLabels and FairEdges can significantly reduce the network's marginalization with relatively few modifications.

Future works could experiment with FairNet in real use cases – for example, a fair recommender system, helping users go out of their social bubbles, or a system to promote diversity in working groups within a school or a corporation. In this context, FairEdges could narrow its scope to only friends-of-friends, adopting a more "local" search (i.e., the node extended neighborhood rather than the whole network). Moreover, instead of only adding edges, FairEdges could also rewire a set of them.

An interesting use-case scenario could be the depolarization of closed substructures emerging from peer interactions in social media platforms (e.g., echo chambers). Recent work has shown that random nudges can effectively lead to less polarized environments [9,25]. Our approach, which starts with random edge insertion and takes into account attribute distribution, can be seen/exploited as a random nudging mechanism enhanced by knowledge of user preferences (encoded as node labels).

FairNet could also be employed for urban planning – each focal node of a city should have a fair and diverse distribution of facilities in its neighborhood. Moreover, the idea behind FairNet could be further tweaked to take into account communities with different distributions of labels. In this scenario, FairNet should compare the label distribution inside a node's neighborhood to the label distribution of the community the node belongs to.

Last, while the algorithms we presented employ a genetic approach, we believe other solutions are viable. A major drawback of genetic algorithms is their scalability under certain scenarios – increasing the number of genes leads to a longer runtime to find viable solutions. Other optimization techniques could be more scalable and, therefore, more suitable for larger networks. For example, a Generative Adversarial Network could be employed to select the best edges.

Appendix: Understanding the IMS

To conceptualize IMS, we start from the assumption that a fair network with $|L| = 2$ and equal distribution of each l (e.g., a network with 50 men and 50 women) should have the following property for all nodes:

$$\frac{|\mathcal{N}_v^l|}{|\mathcal{N}_v^l| + |\mathcal{N}_v^{\neg l}|} = \frac{1}{2}. \tag{3}$$

In other words, each node's neighborhood should feature an equal distribution of labels. We reckon that higher or lower values imply a form of marginalization of v. The following formula normalizes the score in the range $[-1, 1]$:

$$\frac{2 \cdot |\mathcal{N}_v^l|}{|\mathcal{N}_v^l| + |\mathcal{N}_v^{\neg l}|} - 1 = 0. \tag{4}$$

To generalize the formula to networks with more than two attribute values or unequal distribution of labels (e.g., 50 White people, 30 Black people, and 20 Asian people), we assign a weight to each label. The weight of each value l is denoted ω_l and computed as the complementary of the relative frequency of l in the whole network. This implies that rarer labels weigh more than common ones. Additionally, 1 is subtracted from both the numerator and denominator (with 1 representing the node of which we compute the marginalization).

$$\omega_l = 1 - \frac{|\{v : v \in V \wedge l_v = l\}| - 1}{|V| - 1} \tag{5}$$

Since $\mathcal{N}_v^l$ only includes nodes with label l_v, the weight of each node in $\mathcal{N}_v^l$ is ω_{l_v}. Since $\mathcal{N}_v^{\neg l}$ only includes every $l \in L$ except l_v, its weight can simply be computed as $1 - \omega_{l_v}$. Multiplying $|\mathcal{N}_v^l|$ and $|\mathcal{N}_v^{\neg l}|$ by their respective weights results in the IMS formula (Equation 1), a generalization of Equation 4 quantifying the marginalization of a node v in a network G.

The IMS computes the marginalization of each node v w.r.t. the label's distribution in the node's neighborhood, excluding v itself. v could be included by avoiding subtracting 1 from the numerator and the denominator in the weight formula (Equation 5). However, in this scenario, a marginalization score of -1 could not be obtained; even if v were connected only to nodes with different labels, its neighborhood would still include a node with its label, i.e., v itself. For this reason, we decided not to include v as part of its neighborhood. The proposed weight formula accommodates this exclusion.

Acknowledgments. This work is supported by (i) the European Union – Horizon 2020 Program under the scheme "INFRAIA-01-2018-2019 – Integrating Activities for Advanced Communities", Grant Agreement n.871042, "SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics" (`http://www.sobigdata.eu`); (ii) SoBigData.it which receives funding from the European Union – NextGenerationEU – National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR) – Project: "SoBigData.it – Strengthening the Italian RI for Social Mining and Big Data Analytics" – Prot. IR0000013 – Avviso n. 3264 del 28/12/2021; (iii) EU NextGenerationEU programme under the funding schemes PNRR-PE-AI FAIR (Future Artificial Intelligence Research); (iv) TANGO G.A. 101120763.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Alekh Agarwal et al. A reductions approach to fair classification. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 60–69. PMLR, 2018.

2. N Arias Ramos, MD Calvo Sánchez, T Fernández-Villa, MA Ovalle Perandones, D Fernández García, and P Marqués-Sánchez. Social exclusion of the adolescent with overweight: study of sociocentric social networks in the classroom. *Pediatric Obesity*, 13(10):614–620, 2018.
3. Mahmoudreza Babaei, Przemyslaw Grabowicz, Isabel Valera, Krishna P Gummadi, and Manuel Gomez-Rodriguez. On the efficiency of the information networks in social media. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 83–92, 2016.
4. Bliss et al. An evolutionary algorithm approach to link prediction in dynamic social networks. *Journal of Computational Science*, 5(5):750–764, 2014.
5. Bucur and Iacca. Influence maximization in social networks with genetic algorithms. In *European conference on the applications of evolutionary computation*, pages 379–392. Springer, 2016.
6. Emilio Calvano, Giovanni Immordino, and Annalisa Scognamiglio. What drives segregation? evidence from social interactions among students. *Economics of Education Review*, 90:102290, 2022.
7. Chen et al. Proportionally fair clustering. In *International Conference on Machine Learning*, pages 1032–1041. PMLR, 2019.
8. Chhabra et al. An overview of fairness in clustering. *IEEE Access*, 2021.
9. Christopher Brian Currin, Sebastián Vallejo Vera, and Ali Khaledi-Nasab. Depolarization of echo chambers by random dynamical nudge. *Scientific Reports*, 12(1):9234, 2022.
10. Thomas A DiPrete, Andrew Gelman, Tyler McCormick, Julien Teitler, and Tian Zheng. Segregation in social networks based on acquaintanceship and trust. *American journal of sociology*, 116(4):1234–1283, 2011.
11. Dong et al. Fairness in graph mining: A survey. *arXiv preprint arXiv:2204.09888*, 2022.
12. Mark S Granovetter. The strength of weak ties. *American journal of sociology*, 78(6):1360–1380, 1973.
13. Junming Huang, Alexander J Gates, Roberta Sinatra, and Albert-László Barabási. Historical comparison of gender inequality in scientific careers across countries and disciplines. *Proceedings of the National Academy of Sciences*, 117(9):4609–4616, 2020.
14. Fariba Karimi, Mathieu Génois, Claudia Wagner, Philipp Singer, and Markus Strohmaier. Homophily influences ranking of minorities in social networks. *Scientific reports*, 8(1):1–12, 2018.
15. Fariba Karimi and Marcos Oliveira. On the inadequacy of nominal assortativity for assessing homophily in networks. *arXiv preprint arXiv:2211.10245*, 2022.
16. Fariba Karimi, Marcos Oliveira, and Markus Strohmaier. Minorities in networks and algorithms. *arXiv preprint arXiv:2206.07113*, 2022.
17. Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. A review on genetic algorithm: past, present, and future. *Multimedia tools and applications*, 80:8091–8126, 2021.
18. Padmavathi Kora and Priyanka Yadlapalli. Crossover operators in genetic algorithms: A review. *International Journal of Computer Applications*, 162(10), 2017.
19. David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.
20. Federico Mazzoni, Marta Marchiori Manerba, Martina Cinquini, Riccardo Guidotti, and Salvatore Ruggieri. Genfair: A genetic fairness-enhancing data gener-

ation framework. In *International Conference on Discovery Science*, pages 356–371. Springer, 2023.

21. Ninareh Mehrabi, Fred Morstatter, Nanyun Peng, and Aram Galstyan. Debiasing community detection: the importance of lowly connected nodes. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 509–512, 2019.
22. Micha et al. Proportionally fair clustering revisited. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
23. Virginia Morini, Valentina Pansanella, Katherine Abramski, Erica Cau, Andrea Failla, Salvatore Citraro, and Giulio Rossetti. From perils to possibilities: Understanding how human (and ai) biases affect online fora. *arXiv preprint arXiv:2403.14298*, 2024.
24. Hieu Nguyen and Swapna S Gokhale. Analyzing extremist social media content: a case study of proud boys. *Social Network Analysis and Mining*, 12(1):115, 2022.
25. Ritam Pal, Aanjaneya Kumar, and MS Santhanam. Depolarization of opinions on social networks through random nudges. *Physical Review E*, 108(3):034307, 2023.
26. Dana Pessach et al. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)*, 55(3):1–44, 2022.
27. Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *International conference on parallel problem solving from nature*, pages 1081–1090. Springer, 2008.
28. Tahleen Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. Fairwalk: Towards fair graph embedding. 2019.
29. Akrati Saxena, George Fletcher, and Mykola Pechenizkiy. Fairsna: Algorithmic fairness in social network analysis. *arXiv preprint arXiv:2209.01678*, 2022.
30. Shubham Sharma et al. Data augmentation for discrimination prevention and bias disambiguation. In *AIES*, pages 358–364. ACM, 2020.
31. Richard Spillman. Solving large knapsack problems with a genetic algorithm. In *1995 IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century*, volume 1, pages 632–637. IEEE, 1995.
32. Ana-Andreea Stoica and Augustin Chaintreau. Fairness in social influence maximization. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 569–574, 2019.
33. Sotiris Tsioutsiouliklis, Evaggelia Pitoura, Panayiotis Tsaparas, Ilias Kleftakis, and Nikos Mamoulis. Fairness-aware pagerank. In *Proceedings of the Web Conference 2021*, pages 3815–3826, 2021.
34. Claudia Wagner, Philipp Singer, Fariba Karimi, Jürgen Pfeffer, and Markus Strohmaier. Sampling from social networks with attributes. In *Proceedings of the 26th international conference on world wide web*, pages 1181–1190, 2017.
35. Wang et al. Information access equality on generative models of complex networks. *Applied Network Science*, 7(1):1–20, 2022.
36. Wang and Saar-Tsechansky. Augmented fairness: An interpretable model augmenting decision-makers' fairness, 2020.